LA-UR-00-889

Title:

Pulse Coupled Neural Network for Image Smoothing and Segmentation

Authors:

Alexei N. Skourikhine

Accepted to:

International Symposium on Computational Intelligence, Kosice, Slovakia, August 30–September 1, 2000.

Los Alamos
NATIONAL LABORATORY

Pulse-Coupled Neural Networks for Image Smoothing and Segmentation

Alexei N. Skourikhine

Safeguards Systems Group, MS E541, Los Alamos National Laboratory

Los Alamos, NM 87545, USA

E-mail: alexei@lanl.gov

Summary: Pulse-coupled neural networks (PCNNs) based on Eckhorn's model of the cat visual cortex find many applications in image processing, including smoothing, segmentation, and edge extraction. These applications are supported by synchronization of neuron activity, where each neuron corresponds to each pixel in the input image. The synchronization relies on similarities in the gray scale and the linking between neurons. The paper reports on an evaluation based on histogram analysis of PCNN applications to smoothing and segmentation of gray-scale images and also considers the effect of feedback from global inhibitory neuron on the segmentation.

Keywords: pulse-coupled neural networks, image processing, segmentation, image smoothing

1 Image Processing Using Pulse-Coupled Neural Networks

A PCNN is a biologically inspired algorithm for image processing [1,2]. It is to a very large extent based on the Eckhorn model of the cat visual cortex [3,4]. The typical neuron of the PCNN is shown in Fig. 1. The equations for a single iteration of the PCNN are

$$F_{ij}[n] = e^{-\alpha_F} F_{ij}[n-1] + S_{ij} + V_F \sum_{kl} m_{ijkl} Y_{kl}[n-1] + I(t)$$

$$L_{ij}[n] = e^{-\alpha_L} L_{ij}[n-1] + V_L \sum_{kl} w_{ijkl} Y_{kl}[n-1]$$

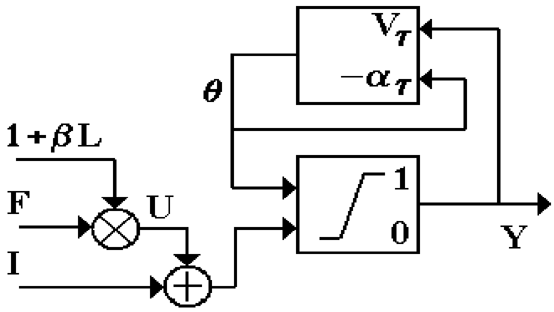

Fig. 1. The basic PCNN neuron.

$$
\begin{aligned}
&U_{ij}[n]=F_{ij}[n](1+\beta L_{ij}[n]) \\
&Y_{ij}[n]=\begin{cases} 1, & if\ U_{ij}[n]>\Theta_{ij}[n] \\ 0, & otherwise \end{cases} \\
&\Theta_{ij}[n]=e^{-\alpha_\Theta}\Theta_{ij}[n-1]+V_\Theta Y_{ij}[n]
\end{aligned}
\qquad (1)
$$

where S is the input signal, F is the feed, L is the link, U is the internal activity, Y is the pulse output, and Θ is the dynamic threshold. The weight matrices M and W are local interconnections and β is the linking constant. I is the inhibition term that is determined by the total activity of the network. The output values of all neurons are summed up, negated, and fed back to each neuron of the neural network.

The basic simplified structure of the pulse-coupled neural network processor for a 2-D input image is shown in Fig. 2. An input gray-scale image is composed of $M\times N$ pixels. This image can be represented as an array of $M\times N$ normalized intensity values. Then the array is fed in at the $M\times N$ inputs of PCNN. If initially all neurons are set to 0, the input results in activation of all of the neurons at a first iteration. The threshold of each neuron, Θ, significantly increases when the neuron fires, then the threshold value decays. When the threshold falls below the respective neuron's potential (U) the neuron fires again, which raises the threshold. The process continues creating binary pulses for each neuron. While this process goes on, neurons encourage their neighbors to fire simultaneously in a way that is supported through interconnections. The firing neurons begin to communicate with their nearest neighbors, which in turn communicate with their neighbors. The result is an autowave that expands from active regions. Thus, if a group of neurons is close to firing, then one neuron can trigger the group. Due to linking between neurons, the pulsing activity of invoked neurons leads to the synchronization between groups of neurons corresponding to subregions of the image that have similar properties and produces a temporal series of binary

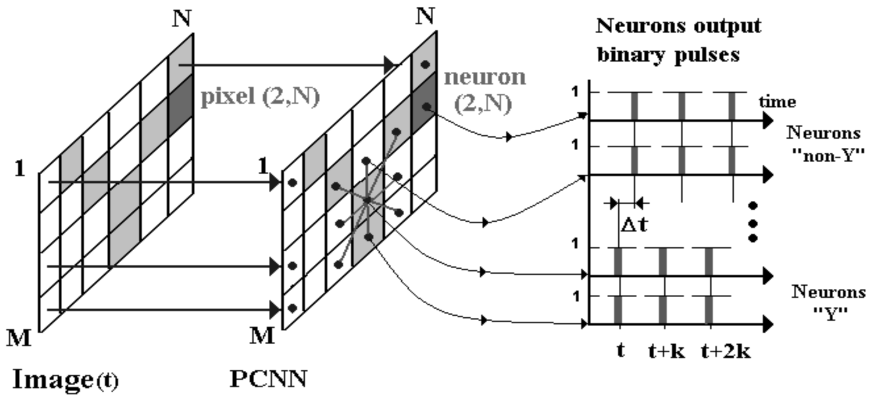

Fig. 2. Image processing using a pulse-coupled neural network**.**

images. This synchronization supports image segmentation.

It is often better to reduce noise in images by filtering before segmentation. PCNNs can also be used for image smoothing, which is achieved through iterated modification of the original intencity levels of an image. Adjustment of a pixel's intensity depends on the history of how in time a pixel's neighbors fire. If a given pixel does not fire simultaneously with the majority of its neighbors, then its intensity is examined. If a pixel fires after the majority of its neighborhood, its intensity is increased, otherwise it is decreased by a small value. This procedure results in both a reduction of the noise contained in the image and the number of pixel intensities represented in the image.

2 Results of Simulation Experiments

By utilizing a PCNN, the gray-scale image (Fig. 3 (a), whose histogram is shown in Fig. 4) is processed by the PCNN algorithm to produce a series of binary images containing the segmentation results. Histograms of the binary images mapped to the original image are shown in Fig. 5 and Fig. 6. Underlying ordering of the PCNN segmentation is obvious from these figures: the brighter the image's region the sooner it fires and is segmented. These histograms also show the difference between the results produced by PCNN without inhibition and PCNN with inhibition. The results of image smoothing are shown in Fig. 3 (b) and Fig. 7.

Our evaluation was performed on a 600 MHz Pentium with 768 Mbytes of RAM and Windows NT. It generally took 0.32 s to produce a single binary image. One run through the whole image required six iterations to have all the image pixels fired in the shown experiment. Thus the minimum total time to process a gray-scale image of 512×512 pixels size is about 2 s.

3 Conclusions

PCNN is useful for image smoothing, segmentation, and edge extraction. However, the algorithm’s speed is crucial, if real-time image processing is desired. Hardware implementation is most appropriate for real-time applications.

The PCNN complexity is the proper setting of the various parameters of the network, such as linking parameters, thresholds, and interconnection matrices. Adjustment of these parameters, depending on the input image, can make PCNN image processing more efficient. At the same time, having adjustable parameters does not solve the problem of interpretation of the results of segmentation. This problem can be solved by the teamwork of the PCNN segmentation module, high-level features grouping modules and recognizer, and specifically by the development of a system with the interactive hierarchy of different layers of processing visual information.

(a) **(b)**

Fig. 3. PCNN smoothing. **(a)** Original image of the size 512×512 pixels. **(b)** Original image after smoothing.

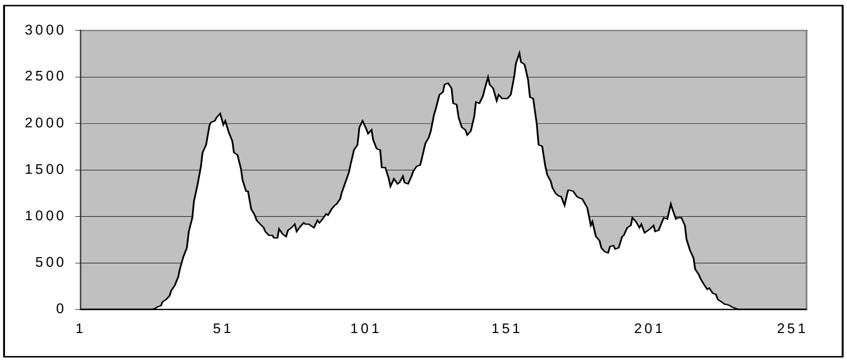

Fig. 4. Histogram of the image shown on Fig. 3.

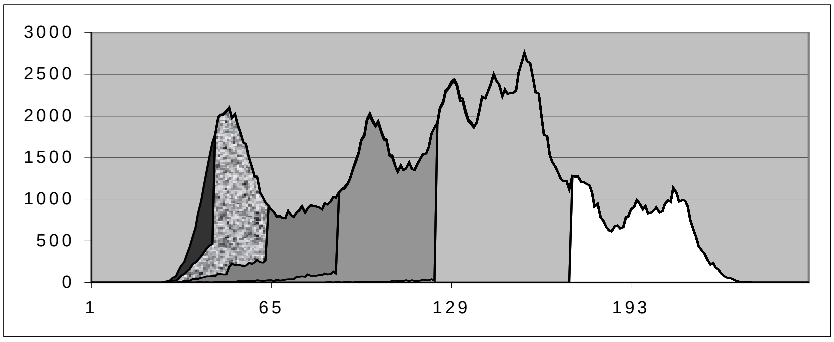

Fig. 5. Six histogram regions (from brightest to darkest) correspond to six binary images produced by PCNN without inhibition.

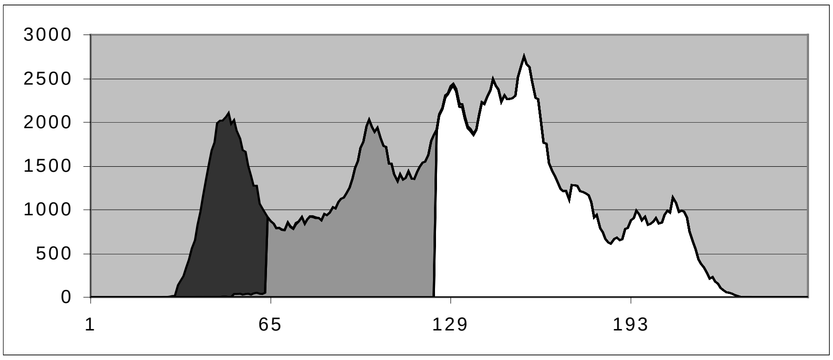

Fig. 6. Three histogram regions (from brightest to darkest) correspond to three binary images produced by PCNN with inhibition.

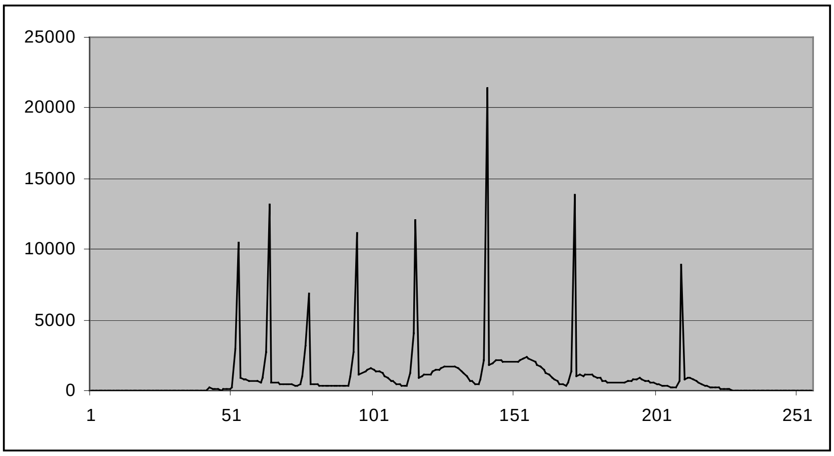

Fig. 7. Histogram of the smoothed image shown in Fig. 3 (b).

References

1. J. L. Johnson, M. L. Padgett. (1999) PCNN models and applications. IEEE Trans. on Neural Networks, **10**, 3, 480–498.

2. T. Lindblad, J. Kinser. (1998) Image Processing using Pulse-Coupled Neural Networks. Springer-Verlag.

3. R. Eckhorn, H. J. Reitboeck, M. Arndt, P. Dicke. (1990) Feature linking via synchronization among distributed assemblies: Simulations of results from cat visual cortex. Neural Comp., **2**, 293–307.

4. M. Stoecker, H. J. Reitboeck, R. Eckhorn. (1996) A neural network for scene segmentation by temporal coding. Neurocomp., **11**, 123–134.